
Exercício Programa II : Integração numérica de Gauss

MAP3121 - MÉTODOS NUMÉRICOS E APLICAÇÕES

PROF. DR. PEDRO PEIXOTO

5 DE JUNHO, 2022

LAURA DO PRADO GONÇALVES PINTO
NºUSP 11819960

MATEUS STANO JUNQUEIRA
NºUSP 11804845

Lista de Figuras

1	Enunciado Exercício Programa II	7
2	Enunciado Exemplo I	9
3	Enunciado Exemplo II	11
4	Enunciado Exemplo III	13
5	Enunciado Exemplo III	16
6	Output total do modo I	20
7	Output total do modo II	21
8	Output dos valores obtidos no modo III no código	22
9	Output de soluções no Wolfram Alpha para primeira parte da solução . .	22
10	Output de soluções no Wolfram Alpha para segunda parte da solução . .	23
11	Output de todos os resultados para o modo IV no código em Python . .	23
12	Resultado obtido na resolução da primeira parte do modo no Wolfram . .	24
13	Resultado obtido na resolução da segunda parte do modo no Wolfram . .	24

Conteúdo

1	Introdução	3
2	Descrição	4
2.1	Análise do problema	4
2.2	Quadratura Gaussiana	4
2.3	Mudança de variável na integração	5
2.4	Integração dupla pelas fórmulas de Gauss	5
3	Implementação do Código	7
3.1	Algoritmo	7
3.2	Metodologia aplicada	7
3.3	Modo 0	7
3.4	Modo I	8
3.5	Modo II	11
3.6	Modo III	13
3.7	Modo IV	16
4	Conclusão	20
4.1	Resultados	20
4.1.1	Modo I	20
4.1.2	Modo II	20
4.1.3	Modo III	21
4.1.4	Modo IV	23
5	Referências	25
	Apêndice A Apêndices	26
A.1	Main.py	26

1 Introdução

Neste relatório é apresentado todo o processo de construção lógica e computacional da resolução do segundo exercício programa proposto para a disciplina MAP3121- Métodos Numéricos e Aplicações assim como os resultados obtidos pelos métodos apresentados e as conclusões tomadas sobre a tarefa proposta.

No exercício proposto trabalha-se a implementação de um algoritmo para resolução de cálculos de integrais duplas utilizando a ferramenta conhecida como fórmulas de integração de Gauss. Para tal tarefa, lança-se mão dos códigos efetuados para a resolução de integrais segundo o método de Gauss em um intervalo específico de $[-1,1]$ com ferramentas adicionais para a mudança de variáveis quando necessário para que este seja válido a quaisquer intervalos desejados. Em seguida, prova-se a confiabilidade do código com a resolução de 4 modos, no caso 4 exercícios propostos ao final da tarefa, os quais dedicam-se a efetuar cálculos em variadas situações.

Este relatório consta a metodologia utilizada para se alcançar tais objetivos além de todos os recursos utilizados para a construção do algoritmo assim como as referentes análises sobre os resultados encontrados.

2 Descrição

2.1 Análise do problema

O problema proposto na tarefa deste primeiro exercício programa consiste em utilizar um método de análise numérica para resolução de integrações por meio das chamadas fórmulas de integração de Gauss. Tais fórmulas consistem em regras de quadratura que aproximam uma integral definida de uma função soma ponderada de valores específicos dentro do domínio da integração. O objetivo central deste exercício programa é a implementação de tal ferramenta para resolução de integrais duplas em regiões R do plano por fórmulas iteradas com o método de Gauss com n nós e pesos fornecidos para o intervalo $[-1, 1]$.

2.2 Quadratura Gaussiana

Uma regra de quadratura Gaussiana consiste em uma fórmula construída para se obter o resultado exato para polinômios de grau $2n-1$ ou menores por meio de uma escolha adequada de nós x_i e pesos w_i para i de 1 à n .

Como mencionado anteriormente, a fórmula de Gauss se traduz em uma soma ponderada cujos pesos possibilitam que por meio dos nós não necessariamente igualmente espalhados chegue-se a uma solução exata para a integral em questão por meio da expressão:

$$\int_{-1}^1 f(x)dx \approx \sum_{i=1}^n w_i f(x_i) \quad (1)$$

O método da quadratura Gaussiana trabalha como escolher tais nós e tais pesos de forma a conseguir a melhor aproximação possível da integral desejada. Como nessa ferramenta trabalha-se com pontos e pesos não iguais em espaçamento como em outros métodos temons n pontos x_i e de tal forma n pesos w_i a serem determinados para resolução completa da integral referida. Assim, trabalhamos com $2n$ graus de liberdade na resolução desse sistema.

Quando desejamos que uma fórmula seja aproximada a um determinado valor, deseja-se que ela seja exata à uma família de funções e, para este caso usaremos os polinômios. Assim, para solução do cálculo desejado aproximaremos a função por um polinômio de grau $2n-1$.

O grau $2n-1$ pode ser explicado pelo simples fato de que para tal equação o número de incógnitas é também igual $2n$, ou seja o mesmo grau de liberdade que a função a qual desejamos aproximá-lo de forma exata.

Para o intervalo de intervalo de integração desejado, a obtenção dos pontos se dá por meio das raízes de um polinômio de Legendre $p_n(x_n)$ por meio dos quais pode-se obter os pesos referentes a cada um deles por meio da equação:

$$c_i = \int_{-1}^1 \prod_{j=1, j \neq i}^n \frac{x - x_j}{x_i - x_j} dx \quad (2)$$

Se temos um espaço vetorial $C[a, b]$ onde estão contidas todas as equações dos polinômios mencionados, no qual é possível definir um produto vetorial tal qual

$$\langle f, g \rangle = \int_a^b f(x)g(x)dx \quad (3)$$

Inicia-se então com as funções polinomiais $1, x, \dots, x^n$, até o n-ésimo polinômio e aplica-se o método de ortogonalização de Gram-Schmidt nessa família criando uma nova leva de n polinômios. Estes polinômios são os polinômios ortogonalizados de Legendre.

2.3 Mudança de variável na integração

A maioria das resoluções e dos cálculos para os nós e pesos no caso discutido até agora pode ser encontrado em tabelas, porém uma questão específica para utilização dessa resolução específica é que o intervalo de integração deve ser $[-1,1]$.

Desta forma, para ser possível realizar esse método será necessário efetuar uma mudança de variável para manipular a integral de valor equivalente de um intervalo qualquer $[a,b]$ para o intervalo desejado $[-1,1]$. Para isso, precisamos pensar em uma função simples que efetue essa equivalência, no caso uma reta que passe por ambos pares de pontos. Assim chega-se em:

$$y = -1 + \frac{2}{b-a}(x-a) \quad (4)$$

trocando x por uma variável u igual a

$$u = -1 + \frac{2}{b-a}(x-a) \quad (5)$$

$$du = \frac{2}{b-a} \cdot dx \quad (6)$$

fazendo as substituições e manipulações necessárias chega-se na seguinte equação para os novos intervalos:

$$x = \frac{b+a}{2} + \frac{b-a}{2} \cdot u \quad (7)$$

Assim a nossa transformação de intervalos será tal que os novos limites e a função no intervalo desejado a ser integrado fica:

$$\int_a^b f(x)dx = \int_{-1}^1 f\left(\frac{b+a}{2} + \frac{b-a}{2} \cdot u\right) \cdot \frac{b-a}{2} du \quad (8)$$

Assim, efetuada a troca demonstrada é possível efetuar a integração pelo método de quadratura gaussiana.

Essa mudança resulta ainda em polinômios exatos até grau cinco visto que para resolução no intervalo $[-1,1]$ pelas fórmulas gaussianas é exata para todo polinômio de grau menor ou igual $2n-1$ e, a equação deduzida para essa operação foi feita para o grau $n=2$ (para dois pontos a e b).

2.4 Integração dupla pelas fórmulas de Gauss

Quando desejamos efetuar uma integral dupla em uma região R do plano, a qual em x está delimitada por duas variáveis $[a,b]$ e em y por duas funções $[c(x),d(x)]$ ou vice-versa. Desta forma podemos usar o cálculo de integrações iteradas por meio das fórmulas de Gauss para n nós tal que:

$$I = \sum_{i=1}^n u_i F(x_i) \quad (9)$$

e

$$F(x_i) = \sum_{j=1}^n v_{ij} f(x_i, y_{ij}) \quad (10)$$

3 Implementação do Código

3.1 Algoritmo

O algoritmo desejado, descrito no enunciado do exercício programa, pede que se implemente um método para o cálculo de integrais duplas por meio das fórmulas de integração de Gauss no intervalo de $[-1,1]$, incluindo métodos para a mudança de variável nos limites de integração para que seja possível a utilização do mesmo método em qualquer intervalo desejado e seja possível efetuar os cálculos com o mesmo método do intervalo fixo mencionado. Em seguida deve-se testar a eficácia e acurácia do algoritmo em 4 exemplos diferentes sobre a resolução de integrais em regiões R do espaço.

De forma geral, é demandado no exercício que sejam criadas funções para a resolução de integrações utilizando o método de quadratura de Gauss, uma para a mudança do intervalo de variável para que o intervalo de integração sempre esteja no determinado $[-1,1]$ e a ampliação da quadratura para sua utilização em integrais duplas no espaço R onde uma variável no espaço está determinado por duas funções e outra por duas variáveis.

Tarefa

Implemente um programa para o cálculo de integrais duplas em regiões R do plano, como as descritas acima, por fórmulas iteradas. Você deverá usar fórmulas de Gauss com n nós para as integrações numéricas. Os nós e os pesos são fornecidos para o intervalo $[-1, 1]$ e o programa deverá fazer os ajustes necessários para outros intervalos. Use *precisão dupla*. No numpy é o padrão e em C use DOUBLE.

Teste o seu programa nos exemplos abaixo usando fórmulas de Gauss com $n = 6, 8$ e 10 . Os nós e os pesos estão no arquivo *dados.txt* (devido à simetria já mencionada, são dados apenas os nós maiores ou iguais a zero).

Figura 1: Enunciado Exercício Programa II

3.2 Metodologia aplicada

O algoritmo aplicado para a resolução desse exercício programa segue a seguinte linha de raciocínio:

- Em primeiro cria-se uma função capaz de efetuar a quadratura de Gauss para resoluções de integrais para o intervalo $[-1,1]$
- Implementa-se um código para efetuação de troca de variável na integração para que qualquer intervalo de integração possa ser resolvido como uma integral definida em $[-1,1]$.
- código para resolução dos exemplos mencionados após o enunciado da tarefa

A seguir será destrinchado cada um destes seguimentos de raciocínio, numerados em modos 0, I, II, III, e IV detalhando as funções, ferramentas e lógicas envolvidas em seu desenvolvimento assim como os resultados obtidos em cada um.

3.3 Modo 0

O assim chamado "Modo 0" neste exercício é o código dedicado a efetuar a quadratura de Gauss para uma integral dupla numa região R do plano, em um intervalo de integração qualquer.

Para executar esse cálculo, criou-se uma função **def integral dupla** apresentada abaixo:

```

1 def integral_dupla(f, a, b, pontos_e_pesos_x, c, d, pontos_e_pesos_y
2 ):
3     """
4     f: função de x e y a ser integrada
5     a: limite inferior do intervalo de integração em x
6     b: limite superior do intervalo de integração em x
7     pontos_e_pesos_x: pontos e pesos para o método de Gauss em x (
8     deve ser da forma [(ponto1, peso1), (ponto2, peso2), ...].)
9     c: limite inferior do intervalo de integração em y (função de x)
10    d: limite superior do intervalo de integração em y (função de x)
11    pontos_e_pesos_y: pontos e pesos para o método de Gauss em y (
12    deve ser da forma [(ponto1, peso1), (ponto2, peso2), ...].)
13
14    """
15
16    I = 0
17    for x_i, w_i in pontos_e_pesos_x:
18        # troca de variavel de x para o intervalo [a,b]
19        x_i = (a + b) / 2 + (b - a) * x_i / 2
20        F = 0
21        for y_ij, w_ij in pontos_e_pesos_y:
22            # troca de variavel de y para o intervalo [c(x_i),d(x_i)]
23            y_ij = (c(x_i) + d(x_i)) / 2 + (d(x_i) - c(x_i)) * y_ij
24            / 2
25            F += w_ij * f(x_i, y_ij) * (d(x_i) - c(x_i)) / 2
26        I += w_i * F * (b - a) / 2
27
28    return I

```

Esta função é basicamente a aplicação das equações elaboradas anteriormente para o código. Inicialmente efetua-se a mudança de variável para que o intervalo de integração esteja sempre em $[-1,1]$. Para tal utiliza-se para ambos intervalos a equação 7 para x e y de forma que em seguida seja possível utilizar as fórmulas 9 e 10 para realizar a quadratura Gaussiana.

O funcionamento do restante do modo é relativamente simples e segue os comentários. Ele consiste em demandar do usuário as entradas requeridas para o funcionamento da função integral dupla, ou seja, recebe e trata as entradas para a função a ser integrada, os limites inferior e superior $[a,b]$ em x da função, os limites inferior e superior em função de x $[c(x),d(x)]$ de y, e os respectivos nós e pesos em x e y.

Vale ressaltar que o número de nós deve ser indicado também nessa parte do programa, sendo este tanto para x quanto para y uma escolha entre 6, 8 ou 10 nós como exigido no enunciado deste exercício programa.

3.4 Modo I

Neste modo trabalha-se a resolução do exemplo I dado após o enunciado da tarefa deste programa, como segue:

Exemplo 1 Calcule os volumes do cubo cujas arestas tem comprimento 1 e do tetraedro com vértices $(0, 0, 0)$, $(1, 0, 0)$, $(0, 1, 0)$ e $(0, 0, 1)$. Você deve obter resultados exatos, exceto por erros de arredondamento (por que?).

Figura 2: Enunciado Exemplo I

Na primeira parte, como estamos calculando o volume de uma forma geométrica específica, ao selecionar modo I já coloca-se que a função a ser integrada será $f(x, y) = 1$, os intervalo $[a, b]$ em x e $[c, d]$ em y são ambos $[0, 1]$. Então, há três opções a se seguir a depender da quantidade de nós escolhidas para a resolução (6, 8 e 10), como segue em anexo:

```

1 elif modo == 1:
2     """Modo 1 – Resolve o Exemplo 1"""
3
4     # Cubo
5
6     print("\n\nCálculo do volume do cubo cujas arestas tem
comprimento 1:\n")
7     print("Função a ser integrada: f(x,y) = 1\n")
8     print("Intervalos de integração: \n a = 0, b = 1, c = 0, d =
1\n")
9     print("Resultados: ")
10    print(
11        "n = 6 —> I = ",
12        integral_dupla(
13            f=lambda x, y: 1,
14            a=0,
15            b=1,
16            pontos_e_pesos_x=n6,
17            c=lambda x: 0,
18            d=lambda x: 1,
19            pontos_e_pesos_y=n6,
20        ),
21    )
22    print(
23        "n = 8 —> I = ",
24        integral_dupla(
25            f=lambda x, y: 1,
26            a=0,
27            b=1,
28            pontos_e_pesos_x=n8,
29            c=lambda x: 0,
30            d=lambda x: 1,
31            pontos_e_pesos_y=n8,
32        ),
33    )
34    print(
35        "n = 10 —> I = ",
36        integral_dupla(
37            f=lambda x, y: 1,

```

```

37         a=0,
38         b=1,
39         pontos_e_pesos_x=n10,
40         c=lambda x: 0,
41         d=lambda x: 1,
42         pontos_e_pesos_y=n10,
43     ),
44     "\n",
45 )
46
47 print("Resultado exato: 1\n")

```

Em seguida elabora-se a mesma linha de raciocínio para resolução da função para um tetraedro, definindo a função a ser integrada $f(x, y) = 1 - x - y$ e os limites em x e y sendo $[0,1]$ e $[0, 1 - x]$, e, em seguida bifurca-se em três entradas possíveis para solução a depender do número de nós escolhidos.

```

1  # Tetraedro
2
3      print(
4          "\n\nCálculo do volume do tetraedro com vertices (0, 0,
5          0), (1, 0, 0), (0, 1, 0) e (0, 0, 1):\n"
6      )
7      print("Função a ser integrada: f(x,y) = 1-x-y\n")
8      print("Intervalos de integração: \n a = 0, b = 1, c = 0, d =
9      1-x\n")
10     print("Resultados: ")
11     print(
12         "n = 6 —> I = ",
13         integral_dupla(
14             f=lambda x, y: 1 - x - y,
15             a=0,
16             b=1,
17             pontos_e_pesos_x=n6,
18             c=lambda x: 0,
19             d=lambda x: 1 - x,
20             pontos_e_pesos_y=n6,
21         ),
22     )
23     print(
24         "n = 8 —> I = ",
25         integral_dupla(
26             f=lambda x, y: 1 - x - y,
27             a=0,
28             b=1,
29             pontos_e_pesos_x=n8,
30             c=lambda x: 0,
31             d=lambda x: 1 - x,
32             pontos_e_pesos_y=n8,
33         ),
34     )

```

```

32     print(
33         "n = 10 —> I = ",
34         integral_dupla(
35             f=lambda x, y: 1 - x - y,
36             a=0,
37             b=1,
38             pontos_e_pesos_x=n10,
39             c=lambda x: 0,
40             d=lambda x: 1 - x,
41             pontos_e_pesos_y=n10,
42         ),
43         "\n",
44     )
45
46     print("Resultado exato: 1/6\n")

```

3.5 Modo II

O segundo modo é dedicado a resolver o segundo exemplo descrito neste exercício como segue:

Exemplo 2 A área A da região no primeiro quadrante limitada pelos eixos e pela curva $y = 1 - x^2$ pode ser obtida por

$$A = \int_0^1 \left[\int_0^{1-x^2} dy \right] dx = \int_0^1 \left[\int_0^{\sqrt{1-y}} dx \right] dy = \frac{2}{3}$$

Calcule numericamente as duas integrais duplas acima e observe os resultados.

Figura 3: Enunciado Exemplo II

Para essa questão a área da região no primeiro quadrante limitada pelos eixos e pela curva $y = 1 - x^2$ já possibilita o modo a assumir a função a ser integrada $f(x, y) = 1$ e os intervalos $[0, 1]$ em x e $[0, 1 - x]$ em y e da mesma forma que no exercício anterior, dedica-se a efetuar três contas diferentes com a única alteração do número de nós escolhidos pelo usuário.

```

1 elif modo == 2:
2     """Modo 2 – Resolve o Exemplo 2"""
3
4     print(
5         "\n\nA area da regioao no primeiro quadrante limitada
6         pelos eixos e pela curva y = 1-x^2:\n"
7     )
8     print("Função a ser integrada: f(x,y) = 1\n")
9     print("Intervalos de integração: \n a = 0, b = 1, c = 0, d =
10     1-x^2\n")
11     print("Resultados: ")
12     print(
13         "n = 6 —> I = ",
14         integral_dupla(

```

```

13         f=lambda x, y: 1,
14         a=0,
15         b=1,
16         pontos_e_pesos_x=n6,
17         c=lambda x: 0,
18         d=lambda x: 1 - x**2,
19         pontos_e_pesos_y=n6,
20     ),
21 )
22 print(
23     "n = 8 —> I = ",
24     integral_dupla(
25         f=lambda x, y: 1,
26         a=0,
27         b=1,
28         pontos_e_pesos_x=n8,
29         c=lambda x: 0,
30         d=lambda x: 1 - x**2,
31         pontos_e_pesos_y=n8,
32     ),
33 )
34 print(
35     "n = 10 —> I = ",
36     integral_dupla(
37         f=lambda x, y: 1,
38         a=0,
39         b=1,
40         pontos_e_pesos_x=n10,
41         c=lambda x: 0,
42         d=lambda x: 1 - x**2,
43         pontos_e_pesos_y=n10,
44     ),
45     "\n",
46 )

```

Em seguida, é utilizado o mesmo método para resolução da integral dupla com ordem invertida: primeiro a integração em x e depois em y. Destaca-se que como essa alteração é não usual, embora a parte impressa em texto no início da resolução indique os valores "invertidos" entre os limites a,b,c,d; no código que o segue a alteração é feita para que o código corra sem maiores problemas. Segue a exemplificação:

```

1 print(
2     "\n\nA area da regioao no primeiro quadrante limitada
   pelos eixos e pela curva x = (1-y)^(1/2):\n"
3 )
4     print("Função a ser integrada: f(x,y) = 1\n")
5     print("Intervalos de integração: \n a = 0, b = (1-y)^(1/2),
   c = 0, d = 1\n")
6     print("Resultados: ")
7     print(
8         "n = 6 —> I = ",
9         integral_dupla(

```

```

10         f=lambda x, y: 1,
11         a= 0,
12         b= 1,
13         pontos_e_pesos_x=n6,
14         c= lambda x : 0,
15         d= lambda x: np.sqrt(1-x) , #necessário alteração da
orientação
16         pontos_e_pesos_y=n6,
17     ),
18 )
19 print(
20     "n = 8 —> I = ",
21     integral_dupla(
22         f=lambda x, y: 1,
23         a=0,
24         b=1,
25         pontos_e_pesos_x=n8,
26         c=lambda x: 0,
27         d=lambda x: np.sqrt(1-x) ,
28         pontos_e_pesos_y=n8,
29     ),
30 )
31 print(
32     "n = 10 —> I = ",
33     integral_dupla(
34         f=lambda x, y: 1,
35         a=0,
36         b=1,
37         pontos_e_pesos_x=n10,
38         c=lambda x: 0,
39         d=lambda x: np.sqrt(1-x) ,
40         pontos_e_pesos_y=n10,
41     ),
42     "\n",
43 )
44
45 print("Resultado exato: 2/3\n")

```

3.6 Modo III

Neste exemplo, como citado abaixo, é dada uma função a qual deve-se integrar para obter o seu volume e em seguida a área da superfície.

Exemplo 3 Considere a superfície descrita por $z = e^{y/x}$, $0.1 \leq x \leq 0.5$, $x^3 \leq y \leq x^2$. Calcule a sua área e o volume da região abaixo dela (a área de uma superfície descrita por $z = f(x, y)$, $(x, y) \in R$ é igual a

$$\iint_R \sqrt{f_x(x, y)^2 + f_y(x, y)^2 + 1} \, dx \, dy).$$

Figura 4: Enunciado Exemplo III

Para a integração do volume, deve-se simplesmente usar o método de integração proposto na função dada z dada nos intervalos de integração também dados, como mostrado a seguir:

```

1 elif modo == 3:
2     """Modo 3 – Resolve o Exemplo 3"""
3     print("Area e volume da superfície descrita por:")
4     print("z = e^(y/x),")
5     print("0.1 <= x <= 0.5,")
6     print("x^3 <= y <= x^2\n")
7
8     print("Obs: x e y devem ser trocados para ordem correta de
9     integração\n")
10
11     # Volume
12
13     print("Cálculo do volume da superfície descrita:\n")
14     print("Função a ser integrada: f(x,y) = e^(y/x)\n")
15     print("Intervalos de integração: \n a = 0.1, b = 0.5, c = x
16     ^3, d = x^2\n")
17     print("Resultados: ")
18     print(
19         "n = 6 —> I = ",
20         integral_dupla(
21             f=lambda x, y: np.e ** (y / x),
22             a=0.1,
23             b=0.5,
24             pontos_e_pesos_x=n6,
25             c=lambda x: x**3,
26             d=lambda x: x**2,
27             pontos_e_pesos_y=n6,
28         ),
29     )
30     print(
31         "n = 8 —> I = ",
32         integral_dupla(
33             f=lambda x, y: np.e ** (y / x),
34             a=0.1,
35             b=0.5,
36             pontos_e_pesos_x=n8,
37             c=lambda x: x**3,
38             d=lambda x: x**2,
39             pontos_e_pesos_y=n8,
40         ),
41     )
42     print(
43         "n = 10 —> I = ",
44         integral_dupla(
45             f=lambda x, y: np.e ** (y / x),
46             a=0.1,
47             b=0.5,

```

```

45         pontos_e_pesos_x=n10,
46         c=lambda x: x**3,
47         d=lambda x: x**2,
48         pontos_e_pesos_y=n10,
49     ),
50     "\n",
51 )

```

Para integração na superfície devemos primeiro chegar na fórmula dada ao final do enunciado, o qual é composto pelas derivadas parciais da função original e após isso integrar a expressão resultante nos intervalos dados na seguinte forma:

```

1  # Superficie
2
3      print("Cálculo da superfície descrita:\n")
4      print(
5          "Função a ser integrada:  $f(x,y) = ( (e^{(x/y)/y})^2 + (-x * e^{(x/y)/y^2})^2 + 1 )^{(1/2)}$ \n"
6      )
7      print("Intervalos de integração: \n a = 0.1, b = 0.5, c = x
8      ^3, d = x^2\n")
9      print("Resultados: ")
10     print(
11         "n = 6 —> I = ",
12         integral_dupla(
13             f=lambda x, y: (
14                 (np.e ** (y / x) / x) ** 2
15                 + (-y * np.e ** (y / x) / x**2) ** 2
16                 + 1
17             )
18             ** (1 / 2),
19             a=0.1,
20             b=0.5,
21             pontos_e_pesos_x=n6,
22             c=lambda x: x**3,
23             d=lambda x: x**2,
24             pontos_e_pesos_y=n6,
25         ),
26     )
27     print(
28         "n = 8 —> I = ",
29         integral_dupla(
30             f=lambda x, y: (
31                 (np.e ** (y / x) / x) ** 2
32                 + (-y * np.e ** (y / x) / x**2) ** 2
33                 + 1
34             )
35             ** (1 / 2),
36             a=0.1,
37             b=0.5,
38             pontos_e_pesos_x=n8,

```



```

37         c=lambda x: x**3,
38         d=lambda x: x**2,
39         pontos_e_pesos_y=n8,
40     ),
41 )
42 print(
43     "n = 10 —> I = ",
44     integral_dupla(
45         f=lambda x, y: (
46             (np.e ** (y / x) / x) ** 2
47             + (-y * np.e ** (y / x) / x**2) ** 2
48             + 1
49         )
50         ** (1 / 2),
51         a=0.1,
52         b=0.5,
53         pontos_e_pesos_x=n10,
54         c=lambda x: x**3,
55         d=lambda x: x**2,
56         pontos_e_pesos_y=n10,
57     ),
58     "\n",
59 )

```

3.7 Modo IV

Exemplo 4 Considere uma região fechada R do plano xy e seja γ uma reta no mesmo plano que não intercepta o interior de R . O volume V do sólido de revolução obtido da rotação da região R em torno de γ é igual a

$$V = 2\pi \iint_R d_\gamma(x, y) dx dy$$

onde $d_\gamma(x, y)$ é a distância do ponto (x, y) à reta γ . Use esta expressão para calcular o volume da calota esférica de altura $\frac{1}{4}$ da esfera de raio 1, e o volume do sólido de revolução obtido da rotação da região, em torno do eixo y , delimitada por $x = 0$, $x = e^{-y^2}$, $y = -1$ e $y = 1$.

Figura 5: Enunciado Exemplo III

Neste último modo, dedica-se a encontrar as soluções em duas etapas: a primeira encontra-se o volume da calota esférica pedida, com $1/4$ de altura do raio e a segunda em calcular o volume do sólido de revolução na rotação no espaço descrito.

Em primeiro, calculando o volume da calota, integra-se a função $f(x,y)=y$ nos intervalos $[3/4, 1]$ - por conta da altura de $1/4$ do raio - e em y de zero à equação da circunferência para a área inicial da esfera utilizada. Então utiliza-se o método de integração:

```

1 elif modo == 4:
2     """Modo 4 – Resolve o Exemplo 4"""
3
4     # Volume da calota esferica

```

```

5      print(
6          "\n\nCalculo do volume da calota esferica de altura 1/4
da esfera de raio 1\n"
7      )
8      print("Função a ser integrada: f(x,y) = y\n")
9      print("Intervalos de integração: \n a = 3/4, b = 1, c = 0, d
= (1-x^2)^(1/2)\n")
10     print("Resultados: ")
11     print(
12         "n = 6 —> I = ",
13         2
14         * np.pi
15         * integral_dupla(
16             f=lambda x, y: y,
17             a=3 / 4,
18             b=1,
19             pontos_e_pesos_x=n6,
20             c=lambda x: 0,
21             d=lambda x: (1 - x**2) ** (1 / 2),
22             pontos_e_pesos_y=n6,
23         ),
24     )
25     print(
26         "n = 8 —> I = ",
27         2
28         * np.pi
29         * integral_dupla(
30             f=lambda x, y: y,
31             a=3 / 4,
32             b=1,
33             pontos_e_pesos_x=n8,
34             c=lambda x: 0,
35             d=lambda x: (1 - x**2) ** (1 / 2),
36             pontos_e_pesos_y=n8,
37         ),
38     )
39     print(
40         "n = 10 —> I = ",
41         2
42         * np.pi
43         * integral_dupla(
44             f=lambda x, y: y,
45             a=3 / 4,
46             b=1,
47             pontos_e_pesos_x=n10,
48             c=lambda x: 0,
49             d=lambda x: (1 - x**2) ** (1 / 2),
50             pontos_e_pesos_y=n10,
51         ),
52         "\n",
53     )

```

54
55

Para a segunda parte integramos a mesma função $f(x,y)$ porém a mudança será nos limites de integração escolhidos, uma vez que estes serão delimitados pelas funções que parametrizam a rotação do sólido formado. Os limites ficam respectivamente $[-1,1]$ em x e $[0, e^{-y^2}]$ em y .

```
1 # Volume do solido de revolucao
2
3     print(
4         "\n\nCalculo do volume do solido de revolucao obtido da
5         rotacao da regioao x=0, x=e^(-y^2), y=-1, y=1 em torno do eixo y\n
6         "
7     )
8     print("Função a ser integrada: f(x,y) = y\n")
9     print("Intervalos de integração: \n a = -1, b = 1, c = 0, d
10    = e^(-y^2)\n")
11    print("Resultados: ")
12    print(
13        "n = 6 —> I = ",
14        2
15        * np.pi
16        * integral_dupla(
17            f=lambda x, y: y,
18            a=-1,
19            b=1,
20            pontos_e_pesos_x=n6,
21            c=lambda x: 0,
22            d=lambda x: np.exp(-(x**2)),
23            pontos_e_pesos_y=n6,
24        ),
25    )
26    print(
27        "n = 8 —> I = ",
28        2
29        * np.pi
30        * integral_dupla(
31            f=lambda x, y: y,
32            a=-1,
33            b=1,
34            pontos_e_pesos_x=n8,
35            c=lambda x: 0,
36            d=lambda x: np.exp(-(x**2)),
37            pontos_e_pesos_y=n8,
38        ),
39    )
40    print(
41        "n = 10 —> I = ",
42        2
43        * np.pi
44        * integral_dupla(
```

```
41         f=lambda x, y: y,
42         a=-1,
43         b=1,
44         pontos_e_pesos_x=n10,
45         c=lambda x: 0,
46         d=lambda x: np.exp(-(x**2)),
47         pontos_e_pesos_y=n10,
48     ),
49     "\n",
50 )
```

4 Conclusão

4.1 Resultados

Seguem abaixo os resultados e comentários sobre os devidos outputs testados para verificação dos códigos apresentados neste exercício programa.

4.1.1 Modo I

Para a verificação deste modo, comparou-se o resultado obtido nas três operações com diferentes número de nós com aquele fornecido no enunciado, o qual está escrito ao final do modo também. No modo um, tanto 6 quanto 8 quanto 10 nós coincidiram com o resultado exato escrito, a única diferença está no grau de precisão, ou seja o número de significativos de cada resposta mas todas chegam na resposta exata do exercício.

```
Modo: 1

Cálculo do volume do cubo cujas arestas tem comprimento 1:

Função a ser integrada:  $f(x,y) = 1$ 

Intervalos de integração:
  a = 0, b = 1, c = 0, d = 1

Resultados:
n = 6 --> I = 1.0
n = 8 --> I = 1.0
n = 10 --> I = 0.9999999999999998

Resultado exato: 1

Cálculo do volume do tetraedro com vertices (0, 0, 0), (1, 0, 0), (0, 1, 0) e (0, 0, 1):

Função a ser integrada:  $f(x,y) = 1-x-y$ 

Intervalos de integração:
  a = 0, b = 1, c = 0, d = 1-x

Resultados:
n = 6 --> I = 0.16666666666666666
n = 8 --> I = 0.16666666666666669
n = 10 --> I = 0.16666666666666666

Resultado exato: 1/6
```

Figura 6: Output total do modo I

4.1.2 Modo II

Para a verificação deste modo, utilizou-se o mesmo método que para o exercício anterior comparou-se o resultado obtido nas três operações com diferentes número de nós com aquele fornecido no enunciado, o qual está indicado na saída ao final do modo também. No modo um, tanto 6 quanto 8 quanto 10 nós coincidiram com o resultado exato mencionado, a única diferença está no grau de precisão, ou seja o número de significativos de cada resposta mas todas chegam na resposta exata do exercício.

Além da confiabilidade do código, é necessário comentar que durante a segunda resolução de duplas integrais é possível observar que a precisão do resultado é ligeiramente inferior à primeira. Isso se deve majoritariamente à utilização da expressão para o limite de x conter uma raiz quadrada ($x = \sqrt{1-y}$), o que acaba propagando uma imprecisão maior que a expressão para a sua integral equivalente que não utiliza ($y = 1 - x^2$)

```

Modo: 2

A area da regioao no primeiro quadrante limitada pelos eixos e pela curva  $y = 1-x^2$ :

Função a ser integrada:  $f(x,y) = 1$ 

Intervalos de integração:
   $a = 0, b = 1, c = 0, d = 1-x^2$ 

Resultados:
n = 6 --> I = 0.6666666666666666
n = 8 --> I = 0.6666666666666667
n = 10 --> I = 0.6666666666666666

A area da regioao no primeiro quadrante limitada pelos eixos e pela curva  $x = (1-y)^{1/2}$ :

Função a ser integrada:  $f(x,y) = 1$ 

Intervalos de integração:
   $a = 0, b = (1-y)^{1/2}, c = 0, d = 1$ 

Resultados:
n = 6 --> I = 0.6670464379156136
n = 8 --> I = 0.6668355801001764
n = 10 --> I = 0.6667560429365088

Resultado exato:  $2/3$ 

```

Figura 7: Output total do modo II

4.1.3 Modo III

Para a conclusão final da confiança no código deste modo, como ao contrário dos anteriores não havia resposta exata disponível, lançou-se mão de calculadoras online utilizadas tipicamente para problemas em cálculo - no caso o Wolfram Alpha.

A seguir observa-se os devidos resultados tanto do código tanto da calculadora. Os outputs são coerentes e podemos considerar o código confiável. Vale a ressalva de que na primeira parte aparece uma pequena parte imaginária na resolução pela calculadora, porém sendo está de ordem de grandeza inferior à 10^{-16} podemos considerar ela desprezível.

```

Modo: 3
Area e volume da superfície descrita por:
z = e^(y/x),
0.1 <= x <= 0.5,
x^3 <= y <= x^2

Cálculo do volume da superfície descrita:

Função a ser integrada: f(x,y) = e^(y/x)

Intervalos de integração:
a = 0.1, b = 0.5, c = x^3, d = x^2

Resultados:
n = 6 --> I = 0.03330556611623718
n = 8 --> I = 0.033305566116232074
n = 10 --> I = 0.033305566116232074

Cálculo da superfície descrita:

Função a ser integrada: f(x,y) = ( (e^(x/y)/y)^2 + (-x*e^(x/y)/y^2)^2 + 1 )^(1/2)

Intervalos de integração:
a = 0.1, b = 0.5, c = x^3, d = x^2

Resultados:
n = 6 --> I = 0.10549788240049787
n = 8 --> I = 0.10549788240051995
n = 10 --> I = 0.10549788240051992

```

Figura 8: Output dos valores obtidos no modo III no código

Computational Inputs:

» function to integrate:

e^{y/x}

» variable 1:

x

» lower limit 1:

0.1

» upper limit 1:

0.5

» variable 2:

y

» lower limit 2:

x³

» upper limit 2:

x²

Compute

Definite integral

$$\int_{0.1}^{0.5} \int_{x^3}^{x^2} \exp\left(\frac{y}{x}\right) dy dx = 0.0333056 - 3.00856 \times 10^{-17} i$$

Figura 9: Output de soluções no Wolfram Alpha para primeira parte da solução

Computational Inputs:

» function to integrate:

» variable 1:

» lower limit 1:

» upper limit 1:

» variable 2:

» lower limit 2:

» upper limit 2:

Compute

Definite integral

$$\int_{0.1}^{0.5} \int_{x^3}^{x^2} \sqrt{1 + \frac{e^{(2y)/x} (x^2 + y^2)}{x^4}} dy dx = 0.105498$$

Figura 10: Output de soluções no Wolfram Alpha para segunda parte da solução

4.1.4 Modo IV

Para este último teste, foram utilizados os mesmos métodos que anteriormente, comparando os resultados obtidos pelo código com aqueles efetuados em ferramentas online para resolução de integrações duplas em regiões do espaço. Mais uma vez observa-se que os resultados estão coerentes e pode-se atestar a confiabilidade do código.

Uma conclusão além da confiabilidade do método é que por se tratar de uma função polinomial e um limite também polinomial o resultado dá exato como o esperado, como mencionado no próprio enunciado.

```

Modo: 4

Calculo do volume da calota esferica de altura 1/4 da esfera de raio 1

Função a ser integrada: f(x,y) = y

Intervalos de integração:
a = 3/4, b = 1, c = 0, d = (1-x^2)^(1/2)

Resultados:
n = 6 --> I = 0.1799870791119152
n = 8 --> I = 0.17998707911191522
n = 10 --> I = 0.17998707911191525

Calculo do volume do solido de revolucao obtido da rotacao da regioao x=0, x=e^(-y^2), y=-1, y=1 em torno do eixo y

Função a ser integrada: f(x,y) = y

Intervalos de integração:
a = -1, b = 1, c = 0, d = e^(-y^2)

Resultados:
n = 6 --> I = 3.7581650328967093
n = 8 --> I = 3.7582492624394384
n = 10 --> I = 3.7582496332093873

```

Figura 11: Output de todos os resultados para o modo IV no código em Python

Computational Inputs:

» function to integrate:

» variable 1:

» lower limit 1:

» upper limit 1:

» variable 2:

» lower limit 2:

» upper limit 2:

[Compute](#)

Definite integral [More digits](#)

$$\int_{\frac{3}{4}}^1 \int_0^{\sqrt{1-x^2}} 2\pi y \, dy \, dx = \frac{11\pi}{192} \approx 0.179987$$

Figura 12: Resultado obtido na resolução da primeira parte do modo no Wolfram

Computational Inputs:

» function to integrate:

» variable 1:

» lower limit 1:

» upper limit 1:

» variable 2:

» lower limit 2:

» upper limit 2:

[Compute](#)

Definite integral [More digits](#)

$$\int_{-1}^1 \int_0^{\exp(-x^2)} 2\pi y \, dy \, dx = \frac{\pi^{3/2} \operatorname{erf}(\sqrt{2})}{\sqrt{2}} \approx 3.75825$$

$\operatorname{erf}(x)$ is the error function

Figura 13: Resultado obtido na resolução da segunda parte do modo no Wolfram

5 Referências

VALLE, Marcos Eduardo. Aula 24 Quadratura Gaussiana. Departamento de Matemática Aplicada Instituto de Matemática, Estatística e Computação Científica Universidade Estadual de Campinas Disponível em : <<https://www.ime.unicamp.br/valle/Teaching/2015/MS211/Aula24.pdf>> Acesso em 31 de maio, 2022.

OLIVEIRA, Maria Luísa Bambozzi de. Integração Numérico. 27 de Outubro, 2010 e 8 de Novembro, 2010. Disponível em <https://sites.icmc.usp.br/marialuisa/cursos201002/integracao_numerica.pdf> . Acesso em 30 de maio, 2022.

Hjorth-Jensen, Morten . Computational Physics Lectures: Numerical integration, from Newton-Cotes quadrature to Gaussian quadrature . Department of Physics, University of Oslo. 23 de agosto, 2017. Disponível em: <<http://compphysics.github.io/ComputationalPhysics/doc/pub/integrate/html/integrate.html>>. Acesso em: 31 maio 2022.

Gauss quadrature formula. Encyclopedia of Mathematics. Disponível em: <http://encyclopediaofmath.org/index.php?title=Gauss_quadrature_formula&oldid=43647> . Acesso em: 30 maio 2022.

INTEGRAÇÃO NUMÉRICA - QUADRATURA DE GAUSS LEGENDRE. Métodos Numéricos. Youtube. 21 de novembro de 2020. 22:33. Disponível em: <<https://www.youtube.com/watch?v=6W5Y2JWQnUg>>. Acesso em: 31 de Maio de 2022.

Calculadora de Integrais Duplas - Wolfram Alpha. Disponível em: <<https://www.wolframalpha.com/input?i=double+integral>>. Acesso em: 2 junho 2022.

NumPy documentation — NumPy v1.22 Manual. Numpy.org. Disponível em: <<https://numpy.org/doc/stable/index.html>>. Acesso em: 2 maio 2022.

Quadratura Gaussiana + Algoritmo em Python. Produção: Sidnei fc. Youtube. 2021. 27:49. Disponível em : <<https://www.youtube.com/watch?v=bu8trr9Qm1Y>> Acesso em 01/06/2022

A Apêndices

A.1 Main.py

```
1  """
2  EP2 – Formulas de Integracao Numerica de Gauss
3
4  Nome: Laura do Prado Gonçalves Pinto
5  NUSP: 11819960
6
7  Nome: Mateus Stano Junqueira
8  NUSP: 11804845
9  """
10
11 import numpy as np
12
13 # pontos e pesos pré estabelecidos
14 # na forma de n = [(x_j,w_j) ,...]
15 n6 = [
16     (0.2386191860831969086305017, 0.4679139345726910473898703),
17     (0.6612093864662645136613996, 0.3607615730481386075698335),
18     (0.9324695142031520278123016, 0.1713244923791703450402961),
19     (-0.2386191860831969086305017, 0.4679139345726910473898703),
20     (-0.6612093864662645136613996, 0.3607615730481386075698335),
21     (-0.9324695142031520278123016, 0.1713244923791703450402961),
22 ]
23 n8 = [
24     (0.1834346424956498049394761, 0.3626837833783619829651504),
25     (0.5255324099163289858177390, 0.3137066458778872873379622),
26     (0.7966664774136267395915539, 0.2223810344533744705443560),
27     (0.9602898564975362316835609, 0.1012285362903762591525314),
28     (-0.1834346424956498049394761, 0.3626837833783619829651504),
29     (-0.5255324099163289858177390, 0.3137066458778872873379622),
30     (-0.7966664774136267395915539, 0.2223810344533744705443560),
31     (-0.9602898564975362316835609, 0.1012285362903762591525314),
32 ]
33 n10 = [
34     (0.1488743389816312108848260, 0.2955242247147528701738930),
35     (0.4333953941292471907992659, 0.2692667193099963550912269),
36     (0.6794095682990244062343274, 0.2190863625159820439955349),
37     (0.8650633666889845107320967, 0.1494513491505805931457763),
38     (0.9739065285171717200779640, 0.0666713443086881375935688),
39     (-0.1488743389816312108848260, 0.2955242247147528701738930),
40     (-0.4333953941292471907992659, 0.2692667193099963550912269),
41     (-0.6794095682990244062343274, 0.2190863625159820439955349),
42     (-0.8650633666889845107320967, 0.1494513491505805931457763),
43     (-0.9739065285171717200779640, 0.0666713443086881375935688),
44 ]
45
46 # Calculo da integral dupla
```

```

47 def integral_dupla(f, a, b, pontos_e_pesos_x, c, d, pontos_e_pesos_y
48 ):
49     """
50     f: função de x e y a ser integrada
51     a: limite inferior do intervalo de integração em x
52     b: limite superior do intervalo de integração em x
53     pontos_e_pesos_x: pontos e pesos para o método de Gauss em x (
54     deve ser da forma [(ponto1, peso1), (ponto2, peso2), ...].)
55     c: limite inferior do intervalo de integração em y (função de x)
56     d: limite superior do intervalo de integração em y (função de x)
57     pontos_e_pesos_y: pontos e pesos para o método de Gauss em y (
58     deve ser da forma [(ponto1, peso1), (ponto2, peso2), ...].)
59
60     """
61
62     I = 0
63     for x_i, w_i in pontos_e_pesos_x:
64         # troca de variavel de x para o intervalo [a,b]
65         x_i = (a + b) / 2 + (b - a) * x_i / 2
66         F = 0
67         for y_ij, w_ij in pontos_e_pesos_y:
68             # troca de variavel de y para o intervalo [c(x_i),d(x_i)]
69             y_ij = (c(x_i) + d(x_i)) / 2 + (d(x_i) - c(x_i)) * y_ij
70             / 2
71             F += w_ij * f(x_i, y_ij) * (d(x_i) - c(x_i)) / 2
72             I += w_i * F * (b - a) / 2
73
74     return I
75
76 def main():
77     print("Escolha um modo:")
78     print("Modo 0 – Calculo de integral dupla para uma função e
79     intervalos quaisquer")
80     print("Modo 1 – Resolve o Exemplo 1")
81     print("Modo 2 – Resolve o Exemplo 2")
82     print("Modo 3 – Resolve o Exemplo 3")
83     print("Modo 4 – Resolve o Exemplo 4")
84     modo = int(input("Modo: "))
85
86     if modo == 0:
87         """Modo 0 – Calculo de integral dupla para uma função e
88         intervalos quaisquer"""
89
90         print("Calculo de integral dupla da forma dydx")
91         f = input("Função f(x,y) a ser integrada: ")
92         f_str = ""
93         for char in f: # Converte a função para uma string
94             utilizavel em eval()
95             if char == "^":

```

```

90         f_str += "**"
91     else:
92         f_str += char
93     f_final = lambda x, y: eval(
94         f
95     ) # Cria uma funcao lambda que recebe x e y e retorna o
valor da função f(x,y)
96
97     a = float(
98         input(
99             "Limite inferior do intervalo de integração da
integral exterior (dx): "
100         )
101     )
102
103     b = float(
104         input(
105             "Limite superior do intervalo de integração da
integral exterior (dx): "
106         )
107     )
108
109     nx = int(input("Número de subintervalos (nós) para x (opções
6, 8 ou 10): "))
110     if nx == 6:
111         nx = n6
112     elif nx == 8:
113         nx = n8
114     elif nx == 10:
115         nx = n10
116     else:
117         raise ValueError("Número de subintervalos inválido")
118
119     c = input(
120         "Limite inferior do intervalo de integração da integral
exterior (dy): "
121     )
122     c_str = ""
123     for char in c: # Converte a função para uma string
utilizavel em eval()
124         if char == "^":
125             c_str += "**"
126         else:
127             c_str += char
128     c_final = lambda x: eval(
129         c_str
130     ) # Cria uma funcao lambda que recebe x e retorna o valor
da função c(x)
131
132     d = input(

```

```

133         "Limite superior do intervalo de integração da integral
exterior (dy): "
134     )
135     d_str = ""
136     for char in d: # Converte a função para uma string
utilizavel em eval()
137         if char == "^":
138             d_str += "**"
139         else:
140             d_str += char
141     d_final = lambda x: eval(
142         d_str
143     ) # Cria uma funcao lambda que recebe x e retorna o valor
da função d(x)
144
145     ny = int(input("Número de subintervalos (nós) para y (opções
6, 8 ou 10): "))
146     if ny == 6:
147         ny = n6
148     elif ny == 8:
149         ny = n8
150     elif ny == 10:
151         ny = n10
152     else:
153         raise ValueError("Número de subintervalos inválido")
154
155     print("\nIntegral da forma dydx:")
156     print(
157         "I = ",
158         integral_dupla(f_final, a, b, nx, c_final, d_final, ny),
159         "\n",
160     )
161
162     elif modo == 1:
163         """Modo 1 – Resolve o Exemplo 1"""
164
165         # Cubo
166
167         print("\n\nCálculo do volume do cubo cujas arestas tem
comprimento 1:\n")
168         print("Função a ser integrada: f(x,y) = 1\n")
169         print("Intervalos de integração: \n a = 0, b = 1, c = 0, d =
1\n")
170         print("Resultados: ")
171         print(
172             "n = 6 —> I = ",
173             integral_dupla(
174                 f=lambda x, y: 1,
175                 a=0,
176                 b=1,

```

```

176         pontos_e_pesos_x=n6,
177         c=lambda x: 0,
178         d=lambda x: 1,
179         pontos_e_pesos_y=n6,
180     ),
181 )
182 print(
183     "n = 8 → I = ",
184     integral_dupla(
185         f=lambda x, y: 1,
186         a=0,
187         b=1,
188         pontos_e_pesos_x=n8,
189         c=lambda x: 0,
190         d=lambda x: 1,
191         pontos_e_pesos_y=n8,
192     ),
193 )
194 print(
195     "n = 10 → I = ",
196     integral_dupla(
197         f=lambda x, y: 1,
198         a=0,
199         b=1,
200         pontos_e_pesos_x=n10,
201         c=lambda x: 0,
202         d=lambda x: 1,
203         pontos_e_pesos_y=n10,
204     ),
205     "\n",
206 )
207
208 print("Resultado exato: 1\n")
209
210 # Tetraedro

```

```

211     print(
212         "\n\nCálculo do volume do tetraedro com vertices (0, 0,
213 0), (1, 0, 0), (0, 1, 0) e (0, 0, 1):\n"
214     )
215     print("Função a ser integrada: f(x,y) = 1-x-y\n")
216     print("Intervalos de integração: \n a = 0, b = 1, c = 0, d =
217 1-x\n")
218     print("Resultados: ")
219     print(
220         "n = 6 → I = ",
221         integral_dupla(
222             f=lambda x, y: 1 - x - y,
223             a=0,
224             b=1,

```

```

223         pontos_e_pesos_x=n6,
224         c=lambda x: 0,
225         d=lambda x: 1 - x,
226         pontos_e_pesos_y=n6,
227     ),
228 )
229 print(
230     "n = 8 → I = ",
231     integral_dupla(
232         f=lambda x, y: 1 - x - y,
233         a=0,
234         b=1,
235         pontos_e_pesos_x=n8,
236         c=lambda x: 0,
237         d=lambda x: 1 - x,
238         pontos_e_pesos_y=n8,
239     ),
240 )
241 print(
242     "n = 10 → I = ",
243     integral_dupla(
244         f=lambda x, y: 1 - x - y,
245         a=0,
246         b=1,
247         pontos_e_pesos_x=n10,
248         c=lambda x: 0,
249         d=lambda x: 1 - x,
250         pontos_e_pesos_y=n10,
251     ),
252     "\n",
253 )
254
255 print("Resultado exato: 1/6\n")
256
257 elif modo == 2:
258     """Modo 2 – Resolve o Exemplo 2"""
259
260     print(
261         "\n\nA area da regioao no primeiro quadrante limitada
262         pelos eixos e pela curva  $y = 1-x^2$ :\n"
263     )
264     print("Função a ser integrada:  $f(x,y) = 1$ \n")
265     print("Intervalos de integração: \n a = 0, b = 1, c = 0, d =
266     1-x^2\n")
267     print("Resultados: ")
268     print(
269         "n = 6 → I = ",
270         integral_dupla(
271             f=lambda x, y: 1,
272             a=0,
273             b=1,

```



```

272         pontos_e_pesos_x=n6,
273         c=lambda x: 0,
274         d=lambda x: 1 - x**2,
275         pontos_e_pesos_y=n6,
276     ),
277 )
278 print(
279     "n = 8 → I = ",
280     integral_dupla(
281         f=lambda x, y: 1,
282         a=0,
283         b=1,
284         pontos_e_pesos_x=n8,
285         c=lambda x: 0,
286         d=lambda x: 1 - x**2,
287         pontos_e_pesos_y=n8,
288     ),
289 )
290 print(
291     "n = 10 → I = ",
292     integral_dupla(
293         f=lambda x, y: 1,
294         a=0,
295         b=1,
296         pontos_e_pesos_x=n10,
297         c=lambda x: 0,
298         d=lambda x: 1 - x**2,
299         pontos_e_pesos_y=n10,
300     ),
301     "\n",
302 )
303
304 print(
305     "\n\nA area da regio no primeiro quadrante limitada
pelos eixos e pela curva  $x = (1-y)^{(1/2)}$ :\n"
306 )
307 print("Função a ser integrada:  $f(x,y) = 1$ \n")
308 print("Intervalos de integração: \n a = 0, b =  $(1-y)^{(1/2)}$ ,
c = 0, d = 1\n")
309 print("Resultados: ")
310 print(
311     "n = 6 → I = ",
312     integral_dupla(
313         f=lambda x, y: 1,
314         a= 0,
315         b= 1,
316         pontos_e_pesos_x=n6,
317         c= lambda x : 0,
318         d= lambda x: np.sqrt(1-x) , #necessário alteração da
orientação
319         pontos_e_pesos_y=n6,

```

```

320         ),
321     )
322     print(
323         "n = 8 → I = ",
324         integral_dupla(
325             f=lambda x, y: 1,
326             a=0,
327             b=1,
328             pontos_e_pesos_x=n8,
329             c=lambda x: 0,
330             d=lambda x: np.sqrt(1-x),
331             pontos_e_pesos_y=n8,
332         ),
333     )
334     print(
335         "n = 10 → I = ",
336         integral_dupla(
337             f=lambda x, y: 1,
338             a=0,
339             b=1,
340             pontos_e_pesos_x=n10,
341             c=lambda x: 0,
342             d=lambda x: np.sqrt(1-x),
343             pontos_e_pesos_y=n10,
344         ),
345         "\n",
346     )
347
348     print("Resultado exato: 2/3\n")
349
350     elif modo == 3:
351         """Modo 3 – Resolve o Exemplo 3"""
352         print("Area e volume da superfície descrita por:")
353         print("z = e^(y/x),")
354         print("0.1 <= x <= 0.5,")
355         print("x^3 <= y <= x^2\n")
356
357         print("Obs: x e y devem ser trocados para ordem correta de
358         integração\n")
359
360         # Volume
361
362         print("Cálculo do volume da superfície descrita:\n")
363         print("Função a ser integrada: f(x,y) = e^(y/x)\n")
364         print("Intervalos de integração: \n a = 0.1, b = 0.5, c = x
365         ^3, d = x^2\n")
366         print("Resultados: ")
367         print(
368             "n = 6 → I = ",
369             integral_dupla(

```

```

367         f=lambda x, y: np.e ** (y / x),
368         a=0.1,
369         b=0.5,
370         pontos_e_pesos_x=n6,
371         c=lambda x: x**3,
372         d=lambda x: x**2,
373         pontos_e_pesos_y=n6,
374     ),
375 )
376 print(
377     "n = 8 → I = ",
378     integral_dupla(
379         f=lambda x, y: np.e ** (y / x),
380         a=0.1,
381         b=0.5,
382         pontos_e_pesos_x=n8,
383         c=lambda x: x**3,
384         d=lambda x: x**2,
385         pontos_e_pesos_y=n8,
386     ),
387 )
388 print(
389     "n = 10 → I = ",
390     integral_dupla(
391         f=lambda x, y: np.e ** (y / x),
392         a=0.1,
393         b=0.5,
394         pontos_e_pesos_x=n10,
395         c=lambda x: x**3,
396         d=lambda x: x**2,
397         pontos_e_pesos_y=n10,
398     ),
399     "\n",
400 )
401
402 # Superfície

```

```

403     print("Cálculo da superfície descrita:\n")
404     print(
405         "Função a ser integrada:  $f(x,y) = (e^{(x/y)/y})^2 + (-x * e^{(x/y)/y^2})^2 + 1$  )^(1/2)\n"
406     )
407     print("Intervalos de integração: \n a = 0.1, b = 0.5, c = x"
408           "\n^3, d = x^2\n")
409     print("Resultados: ")
410     print(
411         "n = 6 → I = ",
412         integral_dupla(
413             f=lambda x, y: (

```

```

414         + (-y * np.e ** (y / x) / x**2) ** 2
415         + 1
416     )
417     ** (1 / 2),
418     a=0.1,
419     b=0.5,
420     pontos_e_pesos_x=n6,
421     c=lambda x: x**3,
422     d=lambda x: x**2,
423     pontos_e_pesos_y=n6,
424 ),
425 )
426 print(
427     "n = 8 —> I = ",
428     integral_dupla(
429         f=lambda x, y: (
430             (np.e ** (y / x) / x) ** 2
431             + (-y * np.e ** (y / x) / x**2) ** 2
432             + 1
433         )
434         ** (1 / 2),
435         a=0.1,
436         b=0.5,
437         pontos_e_pesos_x=n8,
438         c=lambda x: x**3,
439         d=lambda x: x**2,
440         pontos_e_pesos_y=n8,
441     ),
442 )
443 print(
444     "n = 10 —> I = ",
445     integral_dupla(
446         f=lambda x, y: (
447             (np.e ** (y / x) / x) ** 2
448             + (-y * np.e ** (y / x) / x**2) ** 2
449             + 1
450         )
451         ** (1 / 2),
452         a=0.1,
453         b=0.5,
454         pontos_e_pesos_x=n10,
455         c=lambda x: x**3,
456         d=lambda x: x**2,
457         pontos_e_pesos_y=n10,
458     ),
459     "\n",
460 )
461
462 elif modo == 4:
463     """Modo 4 – Resolve o Exemplo 4"""
464

```

```

465 # Volume da calota esferica
466
467 print(
468     "\n\nCalculo do volume da calota esferica de altura 1/4
da esfera de raio 1\n"
469 )
470 print("Função a ser integrada: f(x,y) = y\n")
471 print("Intervalos de integração: \n a = 3/4, b = 1, c = 0, d
= (1-x^2)^(1/2)\n")
472 print("Resultados: ")
473 print(
474     "n = 6 —> I = ",
475     2
476     * np.pi
477     * integral_dupla(
478         f=lambda x, y: y,
479         a=3 / 4,
480         b=1,
481         pontos_e_pesos_x=n6,
482         c=lambda x: 0,
483         d=lambda x: (1 - x**2) ** (1 / 2),
484         pontos_e_pesos_y=n6,
485     ),
486 )
487 print(
488     "n = 8 —> I = ",
489     2
490     * np.pi
491     * integral_dupla(
492         f=lambda x, y: y,
493         a=3 / 4,
494         b=1,
495         pontos_e_pesos_x=n8,
496         c=lambda x: 0,
497         d=lambda x: (1 - x**2) ** (1 / 2),
498         pontos_e_pesos_y=n8,
499     ),
500 )
501 print(
502     "n = 10 —> I = ",
503     2
504     * np.pi
505     * integral_dupla(
506         f=lambda x, y: y,
507         a=3 / 4,
508         b=1,
509         pontos_e_pesos_x=n10,
510         c=lambda x: 0,
511         d=lambda x: (1 - x**2) ** (1 / 2),
512         pontos_e_pesos_y=n10,
513     ),

```

```

513         "\n",
514     )
515
516     # Volume do solido de revolucao
517
518     print(
519         "\n\nCalculo do volume do solido de revolucao obtido da
520         rotacao da regioao x=0, x=e^(-y^2), y=-1, y=1 em torno do eixo y\n
521         "
522     )
523     print("Função a ser integrada: f(x,y) = y\n")
524     print("Intervalos de integração: \n a = -1, b = 1, c = 0, d
525     = e^(-y^2)\n")
526     print("Resultados: ")
527     print(
528         "n = 6 —> I = ",
529         2
530         * np.pi
531         * integral_dupla(
532             f=lambda x, y: y,
533             a=-1,
534             b=1,
535             pontos_e_pesos_x=n6,
536             c=lambda x: 0,
537             d=lambda x: np.exp(-(x**2)),
538             pontos_e_pesos_y=n6,
539         ),
540     )
541     print(
542         "n = 8 —> I = ",
543         2
544         * np.pi
545         * integral_dupla(
546             f=lambda x, y: y,
547             a=-1,
548             b=1,
549             pontos_e_pesos_x=n8,
550             c=lambda x: 0,
551             d=lambda x: np.exp(-(x**2)),
552             pontos_e_pesos_y=n8,
553         ),
554     )
555     print(
556         "n = 10 —> I = ",
557         2
558         * np.pi
559         * integral_dupla(
560             f=lambda x, y: y,
561             a=-1,
562             b=1,
563             pontos_e_pesos_x=n10,

```

```

560         c=lambda x: 0,
561         d=lambda x: np.exp(-(x**2)),
562         pontos_e_pesos_y=n10,
563     ),
564     "\n",
565 )
566
567
568 if __name__ == "__main__":
569     main()

```